PC Data Conversion: Avoiding the Pitfalls

Daniel R. Bretheim, William M. Mercer, Inc.

I. INTRODUCTION

Data conversion procedures can be tremendous time savers, if the capabilities and limitations of those procedures are fully understood. This paper outlines two approaches for converting commonly used PC data formats into SAS® data sets and vice versa. Potential problem areas, a.k.a. pitfalls, are identified and a recommended solution is offered for each.

II. WORKING WITH THE XBASE FILES

This section describes the use of the DBF procedure for converting a Xbase type file (e.g., dBASE III) to a SAS data set and the conversion of SAS data set to Xbase.

A. Database to SAS Data Set

1. **Basic Procedure** - The DBF procedure is intended to convert dBASE II and dBASE III files to SAS data sets. The conversion process will produce a SAS data set that is different in several respects from the original dBASE file. These differences are outlined in the table below.

Characteristic	From dBASE	To SAS
Numeric variables	Character form	Numeric form
Character variables	Character form	Character form (max length 200)
Logical variables	Logical T or F	Character form (length 1)
Date variables	Date format	SAS date variable
Memo fields	Character form	Not converted
Variable name length	10	8
Missing value	No designated value	Zero's

Example: Conversion of a dBASE III File to a SAS Data Set

```
LIBNAME keep 'c:\dataout\';
PROC DBF DB3=data1 OUT=keep.newfile;
RUN;
```

These statements will convert a dBASE III file called **data1.dbf** to a permanent SAS data set called **newfile.ssd**. PROC DBF assumes that the dBASE filename extension is "dbf" and that the file is in the current directory. If the extension is other than "dbf" or if the dBASE file is not in the current directory a FILENAME statement must be used, as shown below.

```
LIBNAME keep 'c:\dataout\';
FILENAME new 'c:\datain\data1.db^';
PROC DBF DB3=new OUT=keep.newfile;
RUN;
```

2. **Results** - Before and after data attributes are displayed below.

File Structure for DATA1.DBF

Structure for database: c:data1.dbf
Number of data records: 5
Date of last update: 07/08/92

Field	Field Name	Type	Width	Dec
1	FIELD1	Character	2	
2	FIELD2	Numeric	5	3
3	FIELD3	Date	8	
4	FIELD4	Logical	1	
5	FIELD5	Memo	10	
6	FIELD6	Character	210	
7	VARIABLE7	Character	5	
8	VARIABLE8	Character	5	
	Total		247	

Contents of NEWFILE.SSD

Data Set Name: KEEP.NEWFILE Type:
Observations: 5 Record Len: 228
Variables: 6
Label:

-- Alphabetic List of Variables and Attributes --

#	Variable	Type	Len	Pos	Format	Label
1	FIELD1	Char	2	4	2.	
2	FIELD2	Num	8	6	5.3	
3	FIELD3	Num	8	14	DATE9.	
4	FIELD4	Char	1	22	1.	
5	FIELD6	Char	200	23	200.	
6	VARIABLE	Char	5	223	5.	

3. **Pitfalls**

a. Any character variable of a length greater than 200 is truncated to 200. **Solution:** Modify the dBASE file structure to redefine field widths to a maximum of 200.

b. dBASE allows 10-character variables names. Variable names will be truncated to a length of 8 when converting a dBASE file to a SAS data set. **Solution:** Either modify the dBASE file structure to shorten to field names or make sure the first 8 characters of each field are unique.

c. If the first 8 characters in dBASE field names are

common to each other, only the first field will be converted. **Solution:** Make sure the first 8 characters of each field name are unique.

d. PROC DBF will not convert dBASE files with more than 32,767 records. **Solution:** Split the dBASE file into several smaller files. Then use PROC DBF to convert the fields into SAS data sets and append them back into a single data set.

e. Records marked for deletion will be translated to the SAS data set. **Solution:** PACK the dBASE file before translating.

f. When converting dBASE IV files to SAS data sets, several potential problems exist.

1) dBASE IV supports up to 255 variables, while dBASE III only supports 128 variables. Since the DB3 option of PROC DBF is used to convert the dBASE IV file, SAS is expecting no more than 128 variables. **Solution:** To circumvent this problem, use the Export Tool in dBASE IV to convert the dBASE IV file to a fixed length fields text file, and then use an INFILE and INPUT statement to read in the ASCII text file.

2) SAS will not convert a dBASE IV file containing a MEMO field. By making the dBASE IV memo field compatible with the dBASE III format, the file can be converted to a SAS data set using the DB3 option of PROC DBF (although the MEMO file will still be ignored). **Solution:** Use the DBMEMO3 option of the COPY TO command in dBASE IV version 1.01.

3) dBASE IV FLOAT fields will not be converted. **Solution:** Issue a MODIFY STRUCTURE command in dBASE to change all FLOAT fields to NUMERIC fields.

B. SAS Data Set to Database

1. **Basic Procedure** - The DBF procedure will also convert a SAS data set to a dBASE file. Conversion differences are outlined in the table below.

Characteristic	From SAS	To dBASE
Numeric variables	Numeric form	Character form (length of 16)
Decimal values	Numeric variable with decimal value	Rounds to the nearest whole number
Date variables	SAS date variables	dBASE date variables
Missing value	SAS missing value	Character '9'

Example: Conversion of a SAS data set to a dBASE III File

```
LIBNAME old 'c:\datain\';
FILENAME new 'c:\dataout\data2.dbf';
PROC DBF DB3=new DATA=old.sasfile;
RUN;
```

These statements will convert a SAS data set called **sasfile.ssd** to a dBASE file called **data2.dbf**. A FILENAME statement is used to specify a fileref that names the dBASE file to be created. In this case, the dBASE file will be written to the c:\dataout subdirectory. The FILENAME statement must precede the PROC DBF statement.

2. **Results** - Before and after data attributes are displayed below.

Contents of SASFILE.SSD

– Variables Ordered by Position –

#	Variable	Type	Len	Pos	Label
1	CHAR1	Char	2	4	
2	NUM1	Num	8	6	
3	DATE1	Num	8	14	
4	LG_CHAR	Char	100	22	
5	LGNUM1	Num	8	122	
6	LGNUM2	Num	8	130	
7	LGNUM3	Num	8	138	

File Structure for DATA2.DBF

Structure for database: C:\DATAOUT\DATA2.DBF
Number of data records: 5
Date of last update: 08/04/92

Field	Field Name	Type	Width	Dec	Index
1	CHAR1	Char	2		
2	NUM1	Numeric	16		
3	DATE1	Numeric	16		
4	LG_CHAR	Char	100		
5	LGNUM1	Numeric	16		
6	LGNUM2	Numeric	16		
7	LGNUM3	Numeric	16		
	Total		183		

3. **Pitfalls**

a. PROC DBF rounds decimals to the nearest integer. If a format has not been assigned to numeric variables with decimal values prior to conversion to a dBASE file, the numbers will be rounded to the nearest whole number. **Solution:** A FORMAT statement can be issued in either a data step or in PROC DBF to cause the decimal portion to be translated.

b. If the number of digits in a SAS numeric variable, including the decimal point, exceeds 16, the resulting dBASE numeric field will be filled with character '9'. For example 1.00E-16 (0.0000000000000001) exceeds 16 digits. After conversion, the dBASE field will contain 16 9's

(9999999999999999 or .1000000000E+17).

c. If the SAS data set has any missing values, the corresponding dBASE numeric field is filled with character '9'. **Solution:** Substitute a numeric value (such as -999) for all missing values and then run PROC DBF. After conversion, use the dBASE REPLACE command to reset all 999's to missing.

4. **Revised Approach** - Inclusion of a FORMAT statement with PROC DBF will produce an output file more comparable to the original SAS data set, as shown by comparing the SAS data set contents and the dBASE file structure below.

```
LIBNAME old 'c:\datain\';
FILENAME new 'c:\dataout\data2.dbf';
PROC DBF DB3=new DATA=old.sasfile;
FORMAT NUM1 5.3 DATE1 DATE9. LGNUM2
LGNUM3 16.15;
RUN;
```

File Structure for DATA2F.DBF

Structure for database: C:\DATAOUT\DATA2.DBF
Number of data records: 5
Date of last update: 08/04/92

Field	Field Name	Type	Width	Dec	Index
1	CHAR1	Char	2		
2	NUM1	Numeric	16	3	
3	DATE1	Date	8		
4	LG_CHAR	Char	100		
5	LGNUM1	Numeric	16		
6	LGNUM2	Numeric	16	15	
7	LGNUM3	Numeric	16	15	
Total			175		

C. **Compatibility** - A cursory review of other database software revealed the following:

Will PROC DBF work with FoxPro? Yes, if the FoxPro file format is consistent with dBASE III.

Will PROC DBF work with Foxbase? Yes. The SAS System will convert a Foxbase file to a SAS data set and Foxbase will read a DBF file created by the DBF procedure.

Will PROC DBF work with AlphaFour? Yes. The SAS System will convert an AlphaFour file to a SAS data set and AlphaFour will read a DBF file created by the DBF procedure.

III. WORKING WITH SPREADSHEETS

The Data Interchange Format (DIF) was developed to be used as a common language for data. Software packages such as Lotus 1-2-3 and VisiCalc use the DIF. A DIF file appears on the screen as a long, narrow row of values or labels separated by a CR-LF pair. Numeric variables can be represented in either integer or scientific notation formats.

Many spreadsheet packages include the capability to save or translate a file in DIF. When saved in this format, the DIF procedure can be used to convert the DIF file into a SAS data set. While there are several steps to this process, the end result is conversion of spreadsheet data into a SAS data set.

A. **Spreadsheet to SAS Data Set**

1. **Basic Procedure** - Conversion of a spreadsheet to a SAS data set is a two step process. First, save or translate the spreadsheet to DIF. Second, use PROC DIF to convert the DIF file to a SAS data set. Any spreadsheet that you plan to convert to a SAS data set should be in a tabular form, where all values in a given row or column represent the same type of data. Inconsistent data, such as dashes, blanks, etc. will create problems.

In the example that follows, the DIF file originates from a Lotus 1-2-3 worksheet file. The worksheet is converted to a DIF file using the Lotus 1-2-3 Translate Utility, which creates the DIF file with the same file name as the worksheet file from which it was translated. The DIF file has the file extension DIF.

Original Lotus 1-2-3 Worksheet

SSN	GPA	GRAD_DATE	REMARKS
548-94-7258	3.183	6/19/90	This col is 30 char in length.
523-93-7881	3.549	5/30/91	This col is 30 char in length.
561-38-3512	3.857	6/13/92	This col is 30 char in length.
536-73-8891		6/02/92	This col is 30 char in length.
539-26-0027	3.007	5/21/88	This col is 30 char in length.

Initial Portion of a DIF File

```
Table
0,1
""
VECTORS
0,4
""
TUPLES
0,7
""
DATA
0,0
""
-1,0
BOT
1,0
"SSN"
1,0
"  GPA"
1,0
"GRAD_DATE"
1,0
"     REMARKS"
-1,0
BOT
-1,0
BOT
1,0
"548-94-7258"
0,3.183000000000000E+00
V
```

Example: Conversion of a DIF File to a SAS Data Set

```
LIBNAME keep 'c:\dataout\';
PROC DIF DIF=mydif OUT=keep.newfile;
RUN;
```

These statements will convert a DIF file called **mydif.dif** to a permanent SAS data set called **newfile.ssd**. PROC DIF assumes that the DIF filename extension is "dif" and that the file is in the current directory. If the extension is other than "dif" or if the DIF file is not in the current directory, a FILENAME statement must be used, as shown below.

```
LIBNAME keep 'c:\dataout\';
FILENAME new 'c:\datain\mydif.raw';
PROC DIF DIF=new OUT=keep.newfile;
RUN;
```

PROC DIF assigns the names COL1, COL2,..., COLn to the variables in the output SAS data set. Use the PREFIX= option to specify a prefix to be used in constructing SAS variables names when converting a DIF file to a SAS data set. For example, if PREFIX=VAR, the new variable names will be VAR1, VAR2, ..., VARn.

2. **Results** - The resulting SAS data set is displayed below. Note that PROC DIF converted all seven rows of the original worksheet, thereby corrupting much of the data.

CONTENTS PROCEDURE

Data Set Name: KEEP.NEWFILE Type:
Observations: 7 Record Len: 84
Variables: 4
Label:
– Alphabetic List of Variables and Attributes –

#	Variable	Type	Len	Pos	Format	Label
1	COL1	Char	20	4	20.	
2	COL2	Char	20	24	20.	
3	COL3	Char	20	44	20.	
4	COL4	Char	20	64	20.	

OBS	COL1	COL2
1	SSN	GPA
2		
3	548-94-7258	
4	523-93-7881	
5	561-38-3512	
6	536-73-8891	
7	539-26-0027	

OBS	COL3	COL4
1	GRAD_DATE	REMARKS
2		
3		This col is 30 char
4		This col is 30 char
5		This col is 30 char
6		This col is 30 char
7		This col is 30 char

3. **Pitfalls**

a. Worksheet rows containing inconsistent data will have a corrupting effect on the conversion. **Solution:** Delete these rows before translating the worksheet to a DIF file. Or, if the worksheet contains title or heading lines, you can either delete these rows or use the SKIP= option which will cause PROC DIF to ignore them. For example, the statements below will cause PROC DIF to ignore the first two rows when converting the DIF file to a SAS data set.

```
PROC DIF DIF=mydif SKIP=2 OUT=newfile;
RUN;
```

b. Lotus date values are stored differently than SAS date values. Lotus stores date values as the number of days since January 1, 1900. The SAS System stores date values as the number of days since January 1, 1960. **Solution:** Convert a Lotus date value into a SAS date value by subtracting 21,917 from the Lotus date value.

c. PROC DIF truncates DIF file character variables to 20 characters. **Solution:** In version 6.03, use the PROC DIF LCHAR option that allows conversion of DIF character variables of up to 200 characters. The syntax is shown below:

```
PROC DIF DIF=diffile OUT=sasfile LCHAR;
RUN;
```

4. **Revised Approach** - The approach outlined below produces a SAS data set more comparable to the original worksheet.

```
PROC DIF DIF=mydif SKIP=2 OUT=keep.newfile
LCHAR PREFIX=var;
RUN;
```

Data Set Name: KEEP.NEWFILE Type:
Observations: 5 Record Len: 61
Variables: 4
Label:

– Alphabetic List of Variables and Attributes –

#	Variable	Type	Len	Pos	Format	Label
1	VAR1	Char	11	4	11.	
2	VAR2	Num	8	15		
3	VAR3	Num	8	23		
4	VAR4	Char	30	31	30.	

OBS	VAR1	VAR2	VAR3	VAR4
1	548-94-7258	3.183	33043	This col is 30 char in length.
2	523-93-7881	3.549	33388	This col is 30 char in length.
3	561-38-3512	3.857	33768	This col is 30 char in length.
4	536-73-8891	.	33757	This col is 30 char in length.
5	539-26-0027	3.007	32284	This col is 30 char in length.

B. SAS Data Set to Spreadsheet

1. **Basic Procedure** - Conversion of a SAS data set into a Lotus 1-2-3 spreadsheet is also a two step process.

First, run PROC DIF to convert the SAS data set to a DIF file. Second, use the Lotus 1-2-3 Translate Utility to translate the DIF file to a 1-2-3 worksheet.

Example: Conversion of a SAS data Set to a DIF File

```
LIBNAME old 'c:\datain\';
FILENAME new 'c:\dataout\newdif.dif;
PROC DIF DIF=new DATA=sasfile;
RUN;
```

These statements will convert a SAS data set called **sasfile** to a DIF file called **newdif.dif.** A FILENAME statement is used to specify a fileref that names the DIF file to be created. The FILENAME statement must precede the PROC DIF statement.

The LABELS statement can be used to cause PROC DIF to write the names of the SAS variables as the first row of the DIF file and a row of blanks as the second row of the DIF file. The actual data portion of the DIF file begins in the third row. The syntax is shown below:

```
PROC DIF DIF=diffile DATA=sasfile LABELS; RUN;
```

2. **Results** - Exhibit 1-A displays the original SAS data set. Exhibit 1-B is the initial worksheet produced from the translated DIF file. In the worksheet, the values for NUM1 have been truncated. Also, the values for DATE1 are represented as a SAS date value.

3. **Pitfalls**

a. PROC DIF rounds decimals to integers when it converts a SAS data set to a DIF file. **Solution:** Use a FORMAT statement to assign a format to numeric variables with decimal value prior to conversion.

Note: When converting a permanent SAS data set to a DIF file, I continued to encounter decimal digit truncation even when a FORMAT statement was used in conjunction with the DIF procedure. However, this problem did not exist when converting a temporary SAS data set.

b. SAS date values are stored differently than Lotus date values. The SAS System stores date values as the number of days since January 1, 1960. Lotus stores date values as the number of days since January 1, 1900. **Solution:** Convert a SAS date value into a Lotus date value by adding 21,917 to the SAS date value.

4. **Revised Approach** - Use of the FORMAT statement eliminates the truncation of decimal digits for numeric values.

```
PROC DIF DIF=new DATA=sasfile LABELS;
FORMAT num1 5.3;
RUN;
```

Following translation of the DIF file and reformatting of the DATE1 column, the resulting worksheet appears as displayed in Exhibit 1-C.

C. **Compatibility** - A cursory review of other spreadsheet software revealed the following:

Will PROC DIF work with Quattro Pro? The SAS System cannot read a DIF file created by Quattro Pro. However, Quattro Pro is able to translate a DIF file produced by the DIF procedure.

Will PROC DIF work with Excel? The SAS System cannot read a DIF file created by Excel.

IV. SUMMARY

What is the significance of these pitfalls? Basically, they represent data quality and validity issues that could potentially affect SAS procedure output. For example, truncated numeric values could significantly affect statistical computations. Likewise, the different ways that missing values are represented could also affect statistical computations. The implications of date value storage differences and variable name limitations are also readily apparent. The bottom line is that many of the pitfalls can simply result in a lot of wasted programmer effort from trying to figure out what happened to the data.

In closing, perhaps the cardinal rule of PC data conversion should be to always review PROC CONTENTS output and a portion of the converted file to determine the data conversion impact on each variable.

SAS is a registered trademark or trademark of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are registered trademarks or trademarks of their respective companies.

EXHIBIT 1-A

OBS	CHAR1	NUM1	DATE	LG_CHAR
1	A	1.234	01JAN92	ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789SUGI18NEWYORKCITY9876543210XXX
2	A	1.234	01JAN92	ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789SUGI18NEWYORKCITY9876543210XXX
3	A	.	01JAN92	ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789SUGI18NEWYORKCITY9876543210XXX
4	A	1.234	01JAN92	ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789SUGI18NEWYORKCITY9876543210XXX
5	A	1.234	01JAN92	ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789SUGI18NEWYORKCITY9876543210XXX

EXHIBIT 1-B

CHAR1	NUM1	DATE1	LG_CHAR
A	1	11688	ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789SUGI18NEWYORKCITY9876543210XXX
A	1	11688	ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789SUGI18NEWYORKCITY9876543210XXX
A	NA	11688	ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789SUGI18NEWYORKCITY9876543210XXX
A	1	11688	ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789SUGI18NEWYORKCITY9876543210XXX
A	1	11688	ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789SUGI18NEWYORKCITY9876543210XXX

EXHIBIT 1-C

CHAR1	NUM1	DATE1	LG_CHAR
A	1.234	01JAN92	ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789SUGI18NEWYORKCITY9876543210XXX
A	1.234	01JAN92	ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789SUGI18NEWYORKCITY9876543210XXX
A	NA	01JAN92	ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789SUGI18NEWYORKCITY9876543210XXX
A	1.234	01JAN92	ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789SUGI18NEWYORKCITY9876543210XXX
A	1.234	01JAN92	ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789SUGI18NEWYORKCITY9876543210XXX